

---

## **12. Postrelační databázové systémy**

<b>12.1. Nové oblasti aplikací databázových technologií .....</b>	<b>2</b>
<b>12.2. Objektově-orientované databáze .....</b>	<b>5</b>
<b>12.3. Objektově-relační databáze .....</b>	<b>12</b>
<b>12.4. Další typy databází a SŘBD a trendy rozvoje DB technologie</b>	<b>17</b>
<b>Literatura.....</b>	<b>18</b>

- relační DBS („klasické“) – typická oblast aplikací: mnoho uživatelů s jednoduchými daty (skalární hodnoty atributů, krátké“ záznamy, typicky pevné délky, „,) a manipulacemi,

**Př) Adresa jako atomický typ (znakový řetězec) vs. adresa jako struktura, resp. objekt.**

- nové oblasti aplikací - CAD, CASE, multimediální databáze, informační systémy úřadů (OIS), datové sklady, XML, ... – omezení relačních systémů nevyhovuje.

## **12.1. Nové oblasti aplikací databázových technologií**

- **Návrhové DB (CAD, CASE)**
  - složité objekty, užitečnost rozlišování typu objektu a samotného objektu s jednoznačnou identifikací, vzájemné odkazy pomocí této identifikace, hierarchie objektů - vztahy A\_PART\_OF → složené objekty
  - relativně malý počet výskytů objektů daného typu
  - správa vývoje (verze objektů)

- **Multimediální DB**
  - text, grafika, číslíkově zpracovaný obraz a zvuk
- **Informační systémy úřadů (Office Information Systems)**
  - dokumenty různých typů, vzájemná provázanost objektů, toky dokumentů, plánovací kalendáře apod.
- **Datové sklady**
  - „vícerozměrný“ pohled na data, speciální operace (OLAP – On-line Analytic Processing).
- **XML (Extendable Markup Language)**
  - Standard pro výměnu strukturovaných dat v podobě tzv. XML dokumentu, model dokumentu je výrazně odlišný od relačního modelu dat.
- ***Hlavní požadavky nových aplikačních oblastí***
  - možnost používání složitých datových typů
  - možnost využití výhod objektové orientace
- ***Podpora relačních SŘBD (SQL-92 a PSM)***
  - datový typ BIT VARYING, CHAR VARYING, ... (typy BLOB)
  - uložené procedury

- **Řešení**
  - **přímé využití služeb správy souborů**
  - **mapování na relační SŘBD**
  - **rozšíření OO programovacích jazyků o správu perzistentních objektů**
  - **objektově-orientované SŘBD (OOSŘBD)**
  - **rozšíření relačních SŘBD o složité typy a OO rysy**
  - **→ objektově-relační SŘBD (ORSŘBD)**
  - **další podpora relačních SŘBD (pro datové sklady, multimédia, XML apod.)**
  - **specializované SŘBD a databáze**

## 12.2. Objektově-orientované databáze

- **Objektově-orientovaný datový model**
  - **Struktura objektu**
    - atributy – nejen nabývající skalárních hodnot
    - operace (zprávy)
    - metody
  - **Třídy objektů**
  - **Dědičnost**
    - hierarchie dědičnosti
  - **Polymorfismus**
  - **Identita objektu**
  - **Složené objekty**
    - hierarchie zahrnutí (containment)
- **Perzistentní OO programovací jazyky a OOSŘBD**
  - **Varianty rozšíření jazyků o podporu perzistence a funkcí SŘBD**
    - knihovny pro podporu perzistence
    - integrace prostředků do jazyka

## ➤ Způsoby řešení perzistence

### ▪ Perzistence na úrovni třídy

- deklarace třídy jako perzistentní, zpravidla SŘBD chápe jako „schopné perzistence“

### ▪ Perzistence při vytvoření objektu

- při vytvoření objektu se řekne, zda je objekt perzistentní nebo ne

### ▪ Perzistence označením

- objekt je vytvořen jako přechodný (transient), ale později může být explicitně označen jako perzistentní

### ▪ Perzistence dosažitelností

- několik objektů je deklarovaných jako (kořenové) perzistentní objekty a všechny další objekty přímo či nepřímo odkazované z těchto objektů jsou také perzistentní

## ➤ Uložení perzistentních objektů

- definice tříd v katalogu, objekty (hodnoty atributů) v databázi, metody často v obyčejných souborech

## ➤ Přístup k perzistentním objektům

- **Pojmenováním objektů**

- vhodné jen pro malý počet objektů

- **Podle OID**

- **Uložením do kolekcí a průchodem kolekcí**

- zpravidla podpora několika typů kolekcí (jsou také objekty s operacemi) – množina, multimnožina, seznam apod.

- *extent třídy* – kolekce všech objektů dané třídy s automatickým vkládáním/vyřazením při vytvoření/zrušení objektu dané třídy. To umožňuje pracovat se třídami podobně jako relacemi (objekt odpovídá n-tici relace).

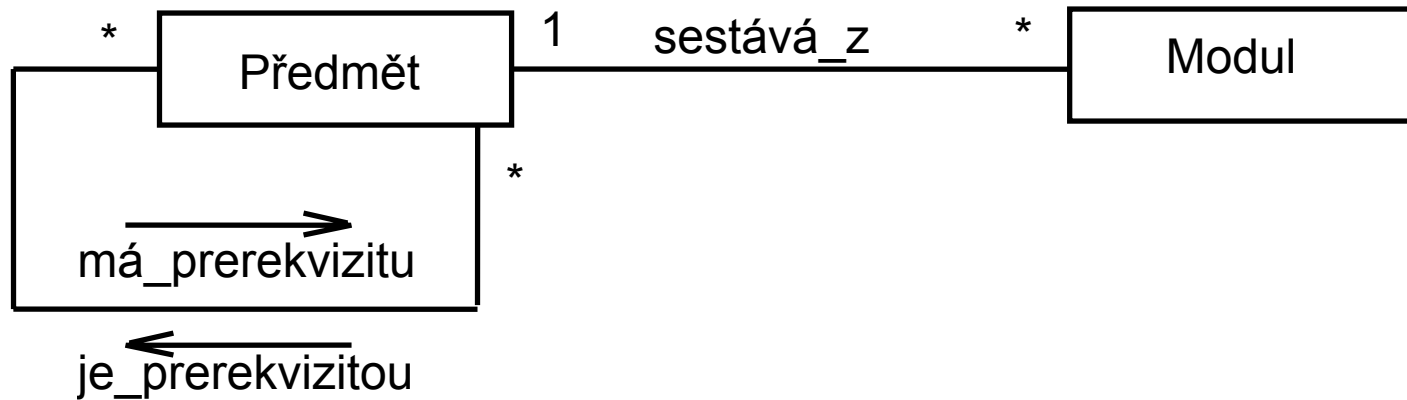
- obvykle podpora všech tří způsobů přístupu (pojmenované zpravidla jen extenty), v extentech zpravidla jen persistentní objekty.

## **Př) GemStone, Jasmine, O2, ODE, Objectivity, ObjectStore, ...**

- manipulačním jazykem zpravidla C++ nebo Smalltalk, navigační programování, případně i podpora pro deklarativní dotazování (např. OSQL v O2, resp. OQL (viz dále))

- **Standard ODMG-93 (verze 1.0 v roce 1993, 3.0 v roce 2000)**
  - **V roce 1991 vznikla skupina ODMG (Object Database Management Group) zahrnující přední společnosti dodávající či vyvíjející OO databázové produkty. Cílem byla snaha o standardizaci v oblasti OO databázových jazyků (viz <http://www.odmg.org/>).**
- **Zahrnuje**
  - **Objektový model dat (neformální definice)**
  - **Jazyk ODL (Object Definition Language) pro definici objektového schématu**
  - **Deklarativní dotazovací jazyk OQL (Object Query Language)**
  - **Vazbu OOSŘBD na C++ (jak implementovat ODL a OQL v prostředí C++, jak psát přenositelné programy pro manipulaci s perzistentními objekty v C++ - tzv. C++OML(Object Manipulation Language))**
  - **Vazbu OOSŘBD na Smalltalk**
  - **Vazbu OOSŘBD na jazyk Java**

Př)



- popis v ODL:

```
interface Predmet
```

```
// vlastnosti typu (třídy):
```

```
( extent Predmety  
  keys cislo)
```

```
// vlastnosti instance (objektu):
```

```
{ attribute String nazev;  
  attribute String cislo;  
  relationship List <Modul> sestava_z_modulu  
    inverse Modul::je_modulem{order_by Modul::cislo};  
  relationship Set <Predmet> ma_prerekvizity
```

```

    inverse Predmet::je_prerekvizitou;
relationship Set <Predmet> je_prerekvizitou
    inverse Predmet::ma_prerekvizity;
// operace instance:
    void nabíika (in Integer semestr) raises
(jiz_nabizeny);
    void zruseni (in Integer semestr) raises
(nenabizeny);
}

```

- **popis v C++ ODL:**

```

class Predmet: public Persistent_Object {
// vlastnosti typu:
    static Ref<Set><Ref<Predmet>>> Predmety
// vlastnosti instance:
    String nazev;
    String cislo;
    List <Ref<Modul>> sestava_z_modulu inverse
Modul::je_modulem;
    Set <Ref<Predmet>> ma_prerekvizity inverse
Predmet::je_prerekvizitou;          ...
}

```

- manipulace - C++ s využitím knihoven zabudovaných tříd
- dotazovací jazyk (OQL) – umožňuje definovat pojmenovaný dotaz  $q$  jako hodnotu výrazu  $e$

```
[define q as] e
```

Př)

**Najdi předmět s názvem „Databázové systémy“.**

```
define dsi as
```

```
  select x
  from x in Predmety
  where x.nazev="Databázové systémy"
```

**Jaké prerekvizity (název předmětu, číslo předmětu) má předmět s názvem „Databázové systémy“.**

```
define prerekv_dsi as
```

```
  select struct(nazev: y.nazev, cislo: y.cislo)
  from x in Predmety, y in x.ma_prerekvizity
  where x.nazev = "Databázové systémy"
```

## 12.3. Objektově-relační databáze

- podpora nenormalizovaných (které nejsou v 1NF) relací, rozšíření relačního modelu o bohatší typový systém a OO rysy
- řada těchto rysů zahrnuta v SQL-99 a podporována předními výrobci relačních systémů (např. Oracle od verze Oracle 8)
- Zanořené relace (nested relations)
  - domény atributů mohou obsahovat buď atomické (skalární) nebo relační (zanořené relace) hodnoty

**Př) Titul (navez, seznam\_autoru, vydavatel, rok\_vydani, klicova\_slova)**

- Složité datové typy
  - Kolekce
    - množiny, multimnožiny, pole
    - v SQL-99 jen pole

**Př)**

**seznam\_autoru VARCHAR(20) ARRAY[10]**

➤ **Rozsáhlé objekty (LOB – Large Objects)**

- **CLOB – rozsáhlá znaková data, BLOB – rozsáhlá binární data**

**Př)**

```
foto BLOB(1MB)
```

- aplikace pracující s rozsáhlým objektem obvykle obdrží jako výsledek SQL dotazu ne celý objekt, ale „lokátor“ pro manipulaci s objektem z hostitelského prostředí

➤ **Uživatелеm definované typy (UDT)**

- jednoduché nebo strukturované (objektové u Oracle)

**Př)**

```
CREATE TYPE TVydavatel AS (  
    nazev VARCHAR(20),  
    pobočka VARCHAR(20))  
CREATE TYPE Ttitul AS (  
    nazev VARCHAR(20),  
    autori VARCHAR(20) ARRAY[10],  
    rok_vydani INTEGER,  
    vydavatel TVydavatel,  
    klicova_slova VARCHAR(30) ARRAY[5])  
CREATE TABLE Titul OF Ttitul /* objektová (Oracle) */
```

- UDT může mít metody

Př)

```
CREATE TYPE TZamestnanec AS (...)  
METHOD zvysPlat (procento FLOAT)
```

```
CREATE METHOD zvysPlat FOR TZamestnanec  
BEGIN  
    SET SELF.plat = SELF.plat+(SELF.plat*procent)/100  
END
```

#### ➤ Konstruktor

- funkce se stejným jménem jako UDT

#### ➤ Dědičnost

- na úrovni typů nebo tabulek (viz generalizace/specializace)
- pouze jednoduchá dědičnost a nepřekrývající se podtypy

Př)

```
CREATE TYPE TOsoba AS (...)  
CREATE TYPE TStudent UNDER TOsoba AS (...)
```

```
CREATE TABLE Osoba OF TOsoba  
CREATE TABLE Student OF TStudent
```

## ➤ Typ reference (REF)

- lze vytvářet explicitní reference řádků tabulek, stejně jako u objektových databází, tj. nejen použitím cizích klíčů

Př)

```
CREATE TYPE TPredmet AS (  
    zkratka CHAR(3),  
    nazev VARCHAR(20),  
    garant REF(TOsoba) SCOPE Osoba)  
CREATE TABLE Predmet OF TPredmet
```

- reference je omezená (v našem případě na řádky tabulky Osoba)
- podle SQL-99 musí mít každá tabulka, na kterou se odkazujeme prostřednictvím typu REF, atribut s identifikátorem řádku, tzv. „sebeodkazující“ (self-referential) atribut

Př)

```
CREATE TABLE Osoba OF TOsoba  
    REF IS oid SYSTEM GENERATED
```

- identifikátor může být generovaný systémem, uživatelem nebo odvozený z primárního klíče

## ➤ Procedury a funkce

- tělo procedur, funkcí a metod může být definováno pomocí procedurálních komponent SQL-99 nebo externě použitím nějakého programovacího jazyka, jako je Java, C nebo C++, případně specializovaného procedurálního jazyka (např. PL/SQL u Oracle)
- procedurální konstrukty SQL-99 dávají SQL vyjadřovací sílu srovnatelnou s programovacími jazyky
- vychází z SQL-92/PSM
- příkazy cyklu WHILE, REPEAT, FOR, IF-THEN-ELSE, přiřazení SET, signalizace výjimečných stavů a zpracování výjimek apod.

### Př) Použití příkazu cyklu FOR pro zpracování výsledku dotazu

```
DECLARE n INTEGR DEFAULT 0;
FOR a AS /* implicitní deklarace kurzoru */
    SELECT stav FROM Ucet
    WHERE pobočka = 'Jánská'
DO
    SET n = n + r.stav
END FOR
```

## **12.4. Další typy databází a SŘBD a trendy rozvoje DB technologie**

- **Logicky orientovaný model - deduktivní databáze (DATALOG)**
- **Modelování prostorových dat - prostorové databáze, GIS**
- **Podpora modelování času - temporální databáze (TSQL)**
- **Podpora aktivity databází - aktivní databáze**
- **Multimediální databáze - podobnostní vyhledávání, efektivní vyhledávání**
- **XML databáze – efektivní ukládání dokumentů a vyhledávání**
- **Systémy na podporu rozhodování - OLAP, datové sklady, získávání znalostí z databází**
- **Integrace a interoperabilita informačních systémů**
- **Přístup k databázím z WWW, Web jako databáze**

## **Literatura**

- 1. Silberschatz, A., Korth H.F, Sudarshan, S.: Database System Concepts. Fourth Edition. McGRAW-HILL. 2001, str. 307 – 360.**
- 2. Pokorný, J.: Databazová abeceda. Science, Veletiny, 1998, str. 39 – 42, 49 – 52, 109 – 118.**